Influence-Based Fair Selection for Sample-Discriminative Backdoor Attacks

Qi Wei[1], Shuo He[1], Jiahan Zhang[3], Lei Feng[2*], Bo An[1,4]

[1]College of Computing and Data Science, Nanyang Technological University, Singapore
[2]Information Systems Technology and Design, Singapore University of Technology and Design, Singapore
[3]Johns Hopkins University, USA
[4]Skywork AI
{qi.wei, shuo.he, boan}@ntu.edu.sg, jzhan423@jh.edu, feng_lei@sutd.edu.sg

Abstract

Backdoor attacks have posed a serious threat in machine learning models, wherein adversaries can poison training samples with maliciously crafted triggers to compromise the victim model. Advanced backdoor attack methods have focused on selectively poisoning more vulnerable training samples, achieving a higher attack success rate (ASR). However, we found that when the manipulation strength of the trigger is constrained to a very small value for imperceptible attacks, they suffer from extremely uneven class-wise ASR due to the unequal selection of instances per class. To solve this issue, we propose a novel backdoor attack method based on Influence-based Fair Selection (IFS), including two objectives: 1) selecting samples that contribute significantly to ASR and 2) ensuring class balance during the selection process. Specifically, we adapt Influence Functions, a classic technique in robust statistics, to evaluate the influence of trigger-embedded training samples on ASR. In this case, training samples contributing to reducing the backdoored test risk could possess higher influence scores. Further, a group-based pruning strategy is designed to avoid calculating the influence on ASR for all training samples, thereby significantly reducing the computational cost. Then, based on the influence score, we design an adaptive thresholding scheme to dynamically select samples with higher influence while maintaining class balance. Extensive experiments on four datasets verify the effectiveness of IFS compared with advanced methods.

1 Introduction

Deep neural networks (DNNs) have achieved remarkable success in various machine learning tasks such as image classification, video understanding, and natural language processing. However, recent works (Gu et al. 2019; Jia, Liu, and Gong 2022; Xia et al. 2022) have revealed that DNNs are vulnerable to backdoor attacks. Specifically, the malicious adversary can elaborately craft backdoor samples (with triggers) and effortlessly distribute them on the Internet. Meanwhile, innocent users or companies might unwittingly crawl these backdoor samples and collect them into a dataset for model training. Once using the poisoned training set for model training, the model will be injected with the trigger. At ordinary times, the infected model would produce normal output, but when encountering the trigger, it would uncontrollably produce adversary-desirable output.

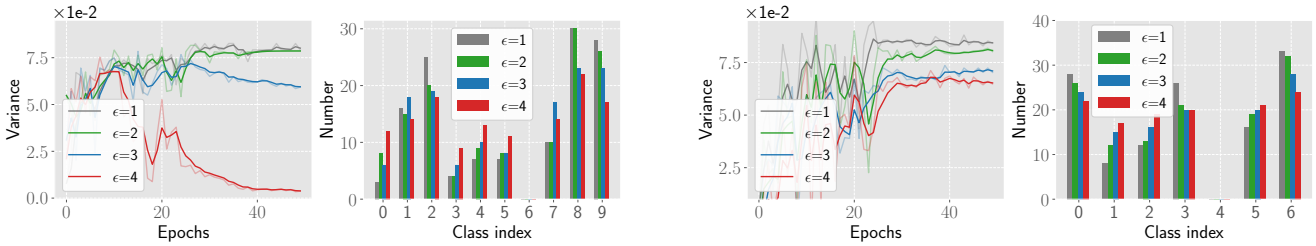
Figure 1: Visualization of backdoored samples with different manipulation strength ϵ, which reflects the visibility of the trigger in a sample. *A smaller value of ϵ is preferred since it enhances stealth.*

The principle of current attack methods is improving the *stealthiness* and *effectiveness* of backdoor attacks. Therefore, they (Gu et al. 2019; Jia, Liu, and Gong 2022) commonly consider only poisoning a small proportion of training samples with visually invisible triggers (controlled by a manipulation strength parameter as shown in Figure 1), which can evade human and machine detection more easily. Furthermore, instead of randomly selecting training samples to be backdoored with heuristic strategies (Gu et al. 2019; Jia, Liu, and Gong 2022), recent methods (Xia et al. 2022; Wu et al. 2023; Xun et al. 2024) proposed to choose representative samples via a pre-designed criterion such as forgetting score (Xia et al. 2022) and representational distance score (Wu et al. 2023), which further improves the effectiveness of backdoor attacks.

However, we found that the attack success rates (ASRs) of these methods are *highly sensitive* to the visibility of the trigger, i.e., the manipulation strength parameter ϵ. For example, when reducing ϵ from 4 (a common setup in existing methods) to 2 (which makes the trigger more imperceptible), current methods have a significant performance degradation (e.g., on ImageNet-10 with a 1% poison rate, the ASRs drops by more than 40% for attack methods FUS (Xia et al. 2022) and RD (Wu et al. 2023)). This observation indicates the limited effectiveness of these attack methods in practical scenarios where high stealthiness is required.

To investigate this phenomenon, we further introduce a new metric, i.e., the variance of class-level ASR, indicating the discrepancy in ASR among different classes, and track the variance of class-level ASR and the number of backdoor samples in each class during the training stage under various values of ϵ. The results in Figure 2 show that when the value of ϵ becomes small, the variance in class-wise ASR is greatly

*Corresponding author



(a) ImageNet-10 (b) Raf_db

Figure 2: Comparison of the variance of class-wise ASR under the patched backdoor with the poisoning rate $r = 1\%$ on two datasets. The (backdoor) targeted label is *class 6* in (a) and *class 4* in (b). The baseline is FUS (Xia et al. 2022). The variance is computed by the formula $\text{Var} = \frac{1}{C}\sum_{c=1}^{C}(a_c - \bar{a})^2$, where C denotes the number of classes, a_c is the ASR on class c and $\bar{a}$ is the overall ASR. A large variance means that backdoor attacks on some categories are successful, while attacks on other categories are less effective. We can observe that *as the value of ϵ decreases, the number of selected samples in each category becomes more imbalanced, leading to a greater variance in class-level ASR.* More results concerning different backdoor types can be found in Appx. B.

large, while the number of backdoor samples in each class is highly imbalanced. This observation reveals that unfair selection for backdoor samples in each class leads to degraded performance on ASR under a small value of ϵ.

Motivated by this observation, in this paper, we propose a novel backdoor attack method based on **I**nfluence-based **F**air **S**election (IFS) that adapts Influence Function (IF) (Cook and Weisberg 1980; Cook 2000; Koh and Liang 2017) for fair sample selection. The intuition of using IF is a sample with a trigger that contributed to decreasing the backdoored test risk would possess a (positively) greater influence on ASR. Based on this point, we can calculate influence values for all training samples with a trigger on ASR and select those with large influence values to construct the backdoor sample set. However, this procedure is highly computational (Koh and Liang 2017). To alleviate this issue, inspired by the discovery that *backdooring representative samples with more distinctive features will contribute more significantly to ASR*, we propose a group-based pruning strategy that splits the overall training samples into different groups and then computes the influence of the group closest to the class prototype. Furthermore, to ensure fair selection across all classes, we introduce a simple yet effective mechanism: customizing class-aware influence thresholds for different classes rather than using a single global threshold. Specifically, in each round of selection, the class-level thresholds are dynamically adjusted to ensure an equal number of instances are selected from each class. In this way, we can achieve fair sample selection for effective backdoor attacks. Extensive experiments on four benchmarks validate the superiority of IFS compared with state-of-the-art counterparts.

Our contribution can be summarized as three-fold:

- *A meaningful observation.* We reveal that the unfair backdoor sample selection leads to significant performance degradation on ASR under a small value of the manipulation strength.
- *A novel selection strategy for backdoor attacks.* We propose a novel backdoor attack method based on influence-based fair selection that provides data-efficient influence computation and fair backdoor sample selection.
- *Superior performances.* We conduct comprehensive experiments on four benchmarks to validate the superiority of the proposed attack method.

2 Related Work

Backdoor Attacks aim to inject an implicit Trojan into the deep model, inducing the model to generate the attacker-prefer target (Chen et al. 2017; Gu et al. 2019). Specifically, suppose a benign training set, the attacker can fulfill the backdoor attack by manipulating the training samples and implanting the trigger into partial samples. When this poisoned training set is released to downstream consumers, any model trained on this set would be infected (Chen et al. 2017). The risk of backdoor attacks has been demonstrated to be significant across many domains, including face recognition (Wenger et al. 2021; Liang et al. 2024), 3D point cloud classification (Li et al. 2021; Xiang et al. 2021), and others.

Threat Model. We follow the principle of poisoning-based backdoor attacks (Xia et al. 2022; Wu et al. 2023; Bagdasaryan et al. 2020; Jiang et al. 2023), indicating a scenario where the victim trains their DNN model on a dataset provided by the adversary and implicitly includes the backdoor. In the inference phase, the adversary injects the trigger into test samples and evaluates the attack success rate. In this paper, we focus only on the black-box backdoor attack, in which the adversary has no access to any training information (e.g., model structure, training strategies, and so on).

Backdooring Sample Selection Strategy. Recently, many works in backdoor attacks propose selecting attack samples more efficiently, given a poison ratio (Nguyen et al. 2024; Zhu et al. 2023). Early works mainly focus on randomly choosing a small faction of samples and adding the trigger (Gu et al. 2019; Jia, Liu, and Gong 2022). However, this strategy overlooks the fact that different samples in the training set have varying sensitivity to the trigger and, therefore, should be treated discriminately. To address this issue, recent works propose a selective framework that chooses the samples significantly contributing to ASR (Li et al. 2023b; Zhu et al. 2023; Li et al. 2023a; He et al. 2023; Li et al. 2024). For instance, Xia et al. (2022) proposed a framework that selects samples by detecting forgetting events and measuring their significance to backdoor attacks. This forgetting event, motivated by catastrophic forgetting (Kirkpatrick et al. 2017), is defined as a scenario where a sample is correctly classi-

fied by the model and then misclassified in the subsequent training round. Further, the sample with a larger amount of forgetting events would be selected. Wu et al. (2023) proposed a criterion called RD score, which distinguishes the sample with a larger gradient in training from all samples and regards them as the most effective.

Notably, the primary advantage of our method lies in its interpretability compared to other empirical selection methods like FUS and RD. IFS can more effectively identify samples for backdooring by assessing the influence of trigger-embedded samples on the ASR.

Influence Functions (IF), which are widely used for sample selection in various tasks. For example, Wang et al. (2020) propose a data subsampling framework that leverages IF to identify and retain the most influential samples in the training dataset. Similarly, Yang et al. (2022) selectively prune the dataset based on the calculated influence, removing data points that harm the model's generalization. Recently, IF has been adopted in backdoor attacks to analyze how backdoored samples influence model learning. For instance, Cinà et al. (2021) utilize IF and incremental learning to demonstrate that models exhibit abnormal learning curves when trained with backdoored samples, indicating that detecting such curves could help identify potential backdoor attacks.

Compared with current IF-based selection frameworks, our proposed IFS is computationally efficient, which is crucial in backdoor attacks where only around 1% of training samples need to be selected. Typically, IFS only requires calculating the influence scores for less than 10% of the training samples, significantly reducing the computational cost.

3 Preliminaries

Problem definition. Given a benign set $D_N = \{(\boldsymbol{x}, \boldsymbol{y})\}^N$ with size N, we aim to select fewer training samples that can greatly contribute to the attack success rate (ASR) if injected with the backdoor. Suppose a poison rate r, the selected set is written as D_M with size M, where $r = \frac{M}{N}$, and the poisoned set is $\mathcal{P} = \{(\boldsymbol{x}', \boldsymbol{y}') | \boldsymbol{x}' = \mathcal{M}(\boldsymbol{x}, \boldsymbol{t}, \epsilon)\}^M$. Note that $\boldsymbol{y}'$ is the adversary-preferred label, and the example $\boldsymbol{x}'$ was implanted with a trigger $\boldsymbol{t}$ via the formulation $\boldsymbol{x}' = \mathcal{M}(\boldsymbol{x}, \boldsymbol{t}, \epsilon) = \boldsymbol{x} \odot (1-\epsilon) + \boldsymbol{t} \odot \epsilon$, where ϵ denotes the manipulation strength. Eventually, the backdoored training set is $\{D_N \backslash D_M\} \cup \mathcal{P}$.

Once this set is utilized for downstream tasks, the victim model f_θ will update the learnable parameters θ via optimizing the following empirical loss:

$$\mathcal{L} = \mathbb{E}_{\substack{(\boldsymbol{x},\boldsymbol{y}) \\ \in D_N \backslash D_M}} [\ell(f_\theta(\boldsymbol{x}), \boldsymbol{y})] + \mathbb{E}_{\substack{(\boldsymbol{x}',\boldsymbol{y}') \\ \in \mathcal{P}}} [\ell(f_\theta(\boldsymbol{x}'), \boldsymbol{y}')] \quad (1)$$

by stochastic gradient descent, where $\ell(\cdot)$ is a classification loss function (e.g., the cross-entropy loss).

Previous methods demonstrated that a very small percentage of poisoned samples could achieve considerable ASR and mainly focus on reducing the poisoned rate r for stealthiness. However, we found another parameter, the manipulation strength ϵ, which is also critical to stealthiness, is always ignored. The issue of unfair selection with a small value of ϵ (illustrated in Introduction) urgently needs to be addressed.

3.1 Influence Functions

We first briefly review Influence Functions (IF) to help motivate and present our algorithm. IF (Cook and Weisberg 1980; Cook 2000; Koh and Liang 2017) provides a fast yet accurate framework for measuring the parameter change when weighting or perturbing an example $\boldsymbol{z}$ during training. Suppose a training set $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$ with a size of n and a model with learnable parameters θ. Considering a training example $\boldsymbol{z}_\delta = (\boldsymbol{x}_\delta, \boldsymbol{y})$ was perturbed by a small value δ in model training, the new model parameters $\hat{\theta}_\delta$ can be written as $\hat{\theta}_\delta = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^n \ell(\boldsymbol{z}_i, \theta) + \delta \ell(\boldsymbol{z}_\delta, \theta)$.

In (Koh and Liang 2017), IF is utilized to estimate the change of the model's prediction on a test point $\boldsymbol{z}_j$, which is sampled from a distribution Q, via the formula:

$$\begin{aligned} \phi_{ij} &= \phi(\boldsymbol{z}_i, \boldsymbol{z}_j \sim Q) \\ &\triangleq \frac{d\ell_j(\hat{\theta}_\delta)}{d\delta}\Big|_{\delta=0} = -\nabla_\theta \ell(\boldsymbol{z}_j, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta \ell(\boldsymbol{z}_i, \hat{\theta}), \end{aligned} \quad (2)$$

where the Hessian matrix $H_{\hat{\theta}} = \frac{1}{n} \sum_{i=1}^n \nabla_\theta^2 \ell(\boldsymbol{z}_i, \hat{\theta})$ and $\nabla_\theta^2 \ell(\boldsymbol{z}_i, \hat{\theta})$ is the second derivative of the loss at training point $\boldsymbol{z}_i$ with respect to θ.

4 Methodology

Adopting the IF framework to *calculate the impact of a training sample with a trigger on the backdoored test risk* is feasible since the small value δ in Eq. (2) can be represented as $\delta = \boldsymbol{t} \odot \epsilon$ if $\epsilon \to 0$, which aims to add an invisible trigger $\boldsymbol{t}$ to the example. However, a direct application is computationally expensive, requiring approximating the Hessian inversion for each pair of training and test samples.

To solve this issue, we propose an **I**nfluence-based **F**air poison sample **S**election framework, termed **IFS**, for backdoor attacks. The main idea behind IFS is to select samples with the greatest influence on the backdoored test risk, *i.e.*, those that contribute the most to ASR. This framework contains two advantages: 1) higher computation efficiency, which does not need to calculate the influence score between each pair of training and test samples, and 2) class-fair selection, mitigating the issue of unfairness in class-wise ASR under a small value of ϵ.

In what follows, we first give an overview of the IFS framework. Then, we describe the major steps in IFS in detail and provide complexity analyses eventually.

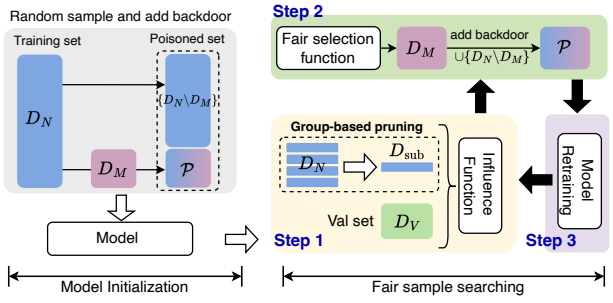


Figure 3: The overview of IFS.

4.1 Overview of IFS

Figure 3 illustrates the overview of IFS, where the framework begins with a phase of model initialization. Given a poison rate, a subset D_M is randomly sampled from the training set D_N and subsequently constructed as the poisoned set $\mathcal{P}$. The model is initialized on a combined dataset comprising $\mathcal{P}$ and $\{D_N \backslash D_M\}$.

Then, fair sample searching, which obeys a greedy search paradigm, alternatively conducts sample search and model training, consisting of *data-efficient influence computation*, *influence-based fair sample selection*, and *model retaining*.

Step 1: *Data-efficient influence computation* is to calculate the influence score for a subset of training samples that are most likely to contribute significantly to ASR. We denote the pruned training set as D_{sub}, satisfying $M < \|D_{\text{sub}}\| < N$, where $\|D_{\text{sub}}\|$ denotes the counts of samples in the set D_{sub}. To achieve this, we propose a group-based data pruning strategy to obtain the subset D_{sub}, which avoids computing the influence for each training sample.

Step 2: *Influence-based fair sample selection* aims to fairly select the backdoor samples from the pruned set D_{sub}. Based on a set of generated influence scores in Step 1, we design class-aware thresholds to dynamically select an equal number of samples from each class.

Step 3: *Model retaining* is to retrain the model on the updated backdoored training set $\{D_N \backslash D_M\} \cup \mathcal{P}$ till convergence to obtain the optimal parameters θ^*.

The pseudocode of IFS can be found in Algorithm 1.

4.2 Data-Efficient Influence Computation

In step 1, we propose, from a data perspective, pruning the scale of training samples to reduce the search space, *i.e.*, switching from searching for D_M within D_N to searching within a subset D_{sub}, which reduces the substantial computation costs caused by the IF framework.

To achieve this, we design a strategy dubbed **group-based pruning**, motivated by the observation that *backdooring samples with more distinctive features will contribute more significantly to ASR*. As shown in Figure 4 (a), we split all samples based on their distance from the class prototype, where samples closer to the prototype better represent the characteristics of that class. We can observe that in the second plot, backdooring the sample in *group 1* (the group closest to the class prototype) probably causes a bigger value of influence, demonstrating that adding the backdoor to these samples will more effectively enhance ASR. Therefore, for lower computation costs, we only need to search the backdoor samples from the group closest to the prototypes.

Specifically, our group-based pruning strategy can be divided into two steps. *i) Class prototypes computation.* In a classification task with C classes, we have C class prototypes for the training set. For the c-th class, its corresponding prototype vector $\mathbf{v}_c$ is obtained by averaging all feature vectors within the class, as follows:

$$\mathbf{v}_c = \frac{1}{\|D^c\|} \sum_{i=1}^{\|D^c\|} g(\boldsymbol{x}_i), \tag{3}$$

where $g(\cdot)$ denotes the feature extractor and D^c represents the set that contains all samples labeled as class c in D_N.

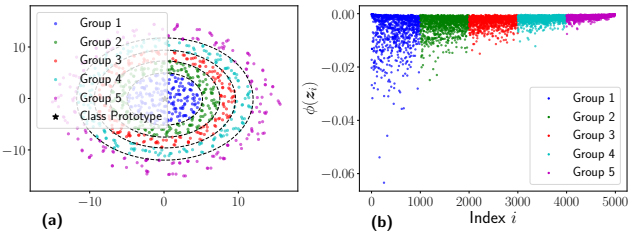


Figure 4: **(a)** Illustration of different groups in one class. Given the class prototype, we split the samples into five equal parts based on the distance between each sample vector and the prototype. **(b) Influence of backdooring different groups on ASR**. We build a 3-layer full-connected model for a binary classification task containing 5,000 positive and 5,000 negative samples, with each example represented as a 768-dimensional vector. The last 10 dimensions are replaced with a specific vector to construct the trigger. We show the influence of samples in *class 0* with a trigger on backdoored test risk. The index of the sample represents its distance to the class prototype - the greater the distance, the larger the index.

ii) Distance-aware group split. We compute the distance between each sample and its prototype vector and split all samples into different groups with the distance. Formally, for each sample $\boldsymbol{z}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$ and $\boldsymbol{z}_i \in D^c$, the distance between this sample and the prototype can be written as

$$\mathrm{d}_i = F_{\text{dis}}(\mathbf{v}_c, g(\boldsymbol{x}_i)), \tag{4}$$

where $F_{\text{dis}}(\cdot)$ denotes a distance metric function, specifically the Euclidean distance, in this paper. Thus, for all samples belonging to D^c, the distance set of them is written as $S_c = \{\mathrm{d}_1, \mathrm{d}_2, ..., \mathrm{d}_{\|D^c\|}\}$. Next, given a hyper-parameter η, we divide D^c into η equal groups. The samples closest to the prototype are designated as *Group 1*, while those farthest from the prototype are designated as *Group* η. Since no more than 2% of the samples are always selected for backdoor attacks, we only need to search within *Group 1*, denoted by

$$G_1^c = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) | i \in \mathrm{Cut}\big(\mathrm{Sort}_{\uparrow}(S_c), \frac{1}{\eta}\big)\}, \tag{5}$$

where $\mathrm{Cut}(\cdot, a)$ is a function that returns indices of the top a proportion of samples, and $\mathrm{Sort}_{\uparrow}(\cdot)$ denotes arranging the values in the input set in ascending order. Applying Eq. (5) over all classes, we have a pruned subset $D_{\text{sub}} \subset D_N$ ($\|D_{\text{sub}}\| = \frac{N}{\eta}$), represented as $D_{\text{sub}} = \{G_1^1, G_1^2, ..., G_1^C\}$.

Then, we calculate the influence of each sample in D_{sub}. Suppose a benign validation set D_V with size U, we can add the backdoor and get its backdoored version $D_V' = \{\boldsymbol{z}_u'\}_{u \in [U]}$. The influence of backdooring the i-th sample in D_{sub} on the average validation loss (or ASR) is computed by aggregating the influence over all examples in D_V':

$$\begin{aligned} \phi_{i, D_{\text{val}}'} &\approx -\frac{1}{U} \sum_{u=1}^{U} \nabla_\theta \ell(\boldsymbol{z}_u', \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta \ell(\boldsymbol{z}_i, \hat{\theta}) \\ &= -\Big[\nabla_\theta \frac{1}{U} \sum_{u=1}^{U} \ell(\boldsymbol{z}_u', \hat{\theta})\Big]^\top H_{\hat{\theta}}^{-1} \nabla_\theta \ell(\boldsymbol{z}_i, \hat{\theta}). \end{aligned} \tag{6}$$

In practice, we resort to the inverse Hessian vector product (IHVP) technique to approximate $H_\theta^{-1} \nabla_\theta \sum_{u=1}^{U} \ell(\boldsymbol{z}_u', \hat{\theta})$

Algorithm 1: Pseudocode of our proposal IFS
(Line 1-2: Model Initialization; Line 3-16: Fair sample searching)

Input: A training set D_N and validation set D_{val}, the search round T, the backdoor rate r, the trigger $\boldsymbol{t}$, the manipulation strength ϵ, the group number η, the deep model f_θ.
Output: A backdoored training set D'_N.

$\mathcal{P} \leftarrow D_M \odot \epsilon\boldsymbol{t}$ // Build the backdoor set on D_M which is drawn from D_N
Initialize f_θ on $\{D_N \backslash D_M\} \cup \mathcal{P}$.
while *search interation* $t \in \{1, ..., T\}$ **do**
  $D_M \leftarrow \emptyset$, $D_{\text{sub}} \leftarrow \emptyset$
  for *class index* $c \in \{1, ..., C\}$ **do**
    $\{G_1^c, ..., G_\eta^c\} \leftarrow D^c$ // Split D^c to η groups, Eq. (3) (4) (5)
    $D_{\text{sub}} \leftarrow G_1^c$ // Add *group 1* of class c to D_{sub}
  $\mathcal{I} \leftarrow \langle D_{\text{sub}}, D_{\text{val}} \rangle$ // Compute influence of samples in D_{sub}, Eq. (6)
  $\{\tau^c\}_{[C]} \leftarrow \langle \{\mathcal{I}^c\}_{[C]}, r \rangle$ // Compute class-level thresholds, Eq. (7)
  $D_M \leftarrow \langle D_{\text{sub}}, \{\mathcal{I}^c, \tau^c\}_{[C]} \rangle$ // Selection via thresholds, Eq. (8)
  for *training epoch* $e \in \{1, ..., E\}$ **do**
    $\mathcal{P} \leftarrow D_M \odot \epsilon\boldsymbol{t}$ // Implant the trigger into the selected set
    Retrain f_θ on $\{D_N \backslash D_M\} \cup \mathcal{P}$ // with Eq. (1)
return the backdoored training set $\{D_N \backslash D_M\} \cup \mathcal{P}$

and avoid direct computation of $H_{\hat{\theta}}^{-1}$. In addition, we adopt the Linear-time Stochastic Second-Order Algorithm (LiSSA) (Agarwal and Bullins 2017) to fast the IHVP.

Eventually, we have a set containing the influence of all samples in D_{sub} on ASR, written as $\mathcal{I} = \{\phi_1, \phi_2, ..., \phi_{\frac{N}{\eta}}\}$.

4.3 Influence-Based Fair Sample Selection

Given a threshold τ, it is natural to select samples whose influence score in $\mathcal{I}$ exceeds τ, to identify those that contribute more significantly to ASR. However, this approach still causes unequal selection when the manipulation strength ϵ is small. To tackle this issue, we provide a simple solution that redesigns the fixed threshold τ to class-level thresholds $\{\tau^1, \tau^2 ..., \tau^C\}$, which directly selects the same quantity of samples in each class.

Specifically, suppose a pre-given backdoor rate r, the number of selected samples per class is $\frac{N}{C} \cdot r$ when fair selection is required, which accounts for the percentage of $\eta \cdot r$ of an individual pruned group G_1. For c-th class, the pruned group is G_1^c and the corresponding influence calculated on Eq. (6) is $\mathcal{I}^c = \{\phi_1, \phi_2, ..., \phi_{\frac{N}{\eta \cdot C}}\}$. Thus, for class c, we can get the class-level dynamic threshold τ^c via the equation:

$$\tau^c = \text{Quantile}\big(\text{Sort}_\downarrow(\mathcal{I}^c), \eta r\big), \tag{7}$$

where $\text{Quantile}(\cdot, a)$ is a function that returns the value at the given quantile a of the set, and $\text{Sort}_\uparrow(\cdot)$ denotes arranging the values in the input set in decreasing order.

Subsequently, we can select samples in G_1^c whose influence scores exceed the threshold τ^c for backdooring. The entirely searched set D_M over all classes is written as

$$D_M \leftarrow \{(\boldsymbol{x}_i, \boldsymbol{y}_i) | \phi_i > \tau^c\}_{i \in \|G_1^c\|}, \forall\, c \in [C]. \tag{8}$$

In this way, samples with a larger contribution to ASR are selected for D_M, while maintaining class balance.

4.4 Complexity Analysis

In this section, we provide the complexity of IFS. Let p denote the number of model parameters. Since the sample dimension is much smaller than the number of model parameters, we omit the sample dimension in the analysis below.

The main complexity arises from the Eq. (6), which involves the computation of the second-order information. In this equation, $[\nabla_\theta \frac{1}{U} \sum_{u=1}^{U} \ell(\boldsymbol{z}'_u, \hat{\theta})]^\top H_{\hat{\theta}}^{-1}$ is fixed, so it only needs to be computed once for different training samples, benefiting reduce the overall running time. Obtaining influence of the sample in the pruned set D_{sub} requires $\mathcal{O}(kp)$ for computing the loss of k training samples, and $\mathcal{O}(Up{+}rjp)$ for the single computation of IHVP via LiSSA, where r, j is two hyper-parameters represented as the recursion depth and the number of recursions, respectively. A larger value of r, j contributes to accurately estimating IHVP. In sum, the computation cost for the influence score is $\mathcal{O}(kp + Up + rjp)$.

Besides, the group pruning and class-fair influence-based selection takes $\mathcal{O}(k \log(k))$, which denotes the complexity of ordinary sorting algorithms. In Algorithm 1, we conduct T times of influence computation and selection, and $T \times E$ times of model training which takes $\mathcal{O}(Np)$.

Eventually, the total complexity of our IFS is $\mathcal{O}(Tp(k + U + rj + EN)) + \mathcal{O}(k \log(k))$. Since $k = \frac{N}{\eta}$, $\frac{1}{\eta} \ll E \ll p$, the complexity can be written as $\tilde{\mathcal{O}}(Tp(U{+}N{+}rj))$, which omits the item of $\mathcal{O}(\log(k))$.

5 Experiments

5.1 Experimental Settings

Datasets. Refer to Wu et al. (2023), we conduct experiments on four datasets, including CIFAR-10 (Krizhevsky, Hinton et al. 2009), ImageNet-10 (Deng et al. 2009), Raf-db (Li, Deng, and Du 2017), and ModelNet40 (Wu et al. 2015), which contains 10, 10, 7, and 40 classes, respectively. More statistical information about datasets is shown in Appx. C.1.
Implementation details. We try varying attack types, including *blended* (Chen et al. 2017) and *patched* (Gu et al. 2019) attacks. For 2D classification tasks, we choose VGG16 as the surrogate model and ResNet18 as the target model. We conduct 10 iterations for searching backdoor samples. After searching in each iteration, we train the model for 70 epochs with a learning rate of 0.01 divided by 10 at the 35, 50-th epoch. We set the training batch size as 128 for both of these three tasks. For 3D point cloud datasets, we adopt PointNet (Qi et al. 2017a) as the surrogate model and PointNet++ (Qi et al. 2017b) as the target model. We conduct 10 iterations for searching. In each iteration, we train the model 50 epochs with a batch size of 128 at a learning rate of 0.01 divided by 10 at the 30-th epoch. All experiments are conducted three times on NVIDIA GTX 3090 GPUs and the average results are reported. More details can be found in Appx. C.1.

For the group number η, we set η to 30 for CIFAR-10, 5 for ImageNet-10 and Raf_db, and 10 for ModelNet40. In addition, we set class 0 as the targeted class for all datasets.

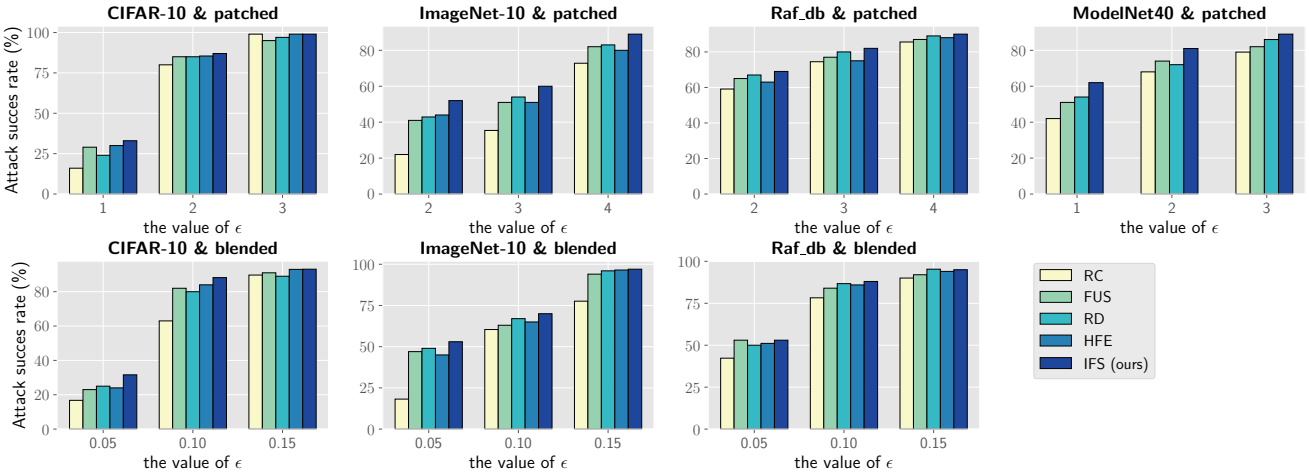


Figure 5: Performance comparison of ASR with **different manipulation strengths** ϵ given a fixed poisoning rate $r = 1\%$. Note that HFE is not suitable for 3D point cloud tasks.

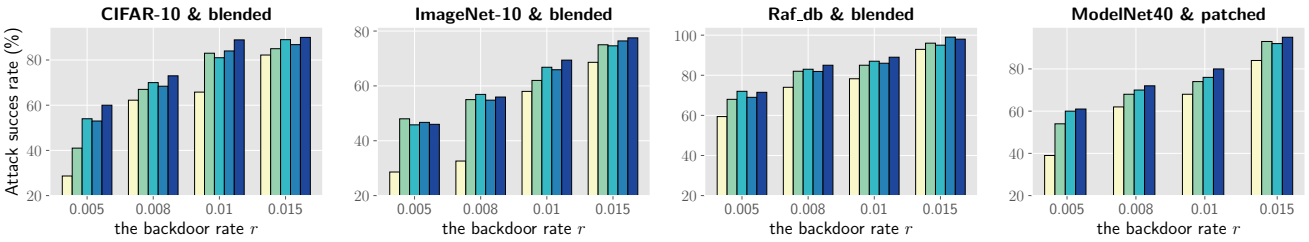


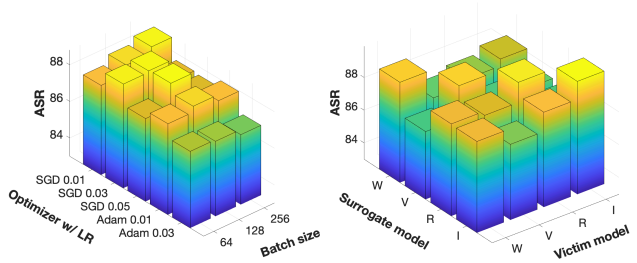
Figure 6: Performance comparison of ASR with **different backdoor rates** r. For the blended attacks (three 2D image tasks), $\epsilon = 0.1$. For the patched attack (3D point cloud task), $\epsilon = 2$.

Baselines. We compared our proposal IFS with the following advancing methods. 1) **Random search (RS)**, randomly selecting samples from untargeted classes with a rate of r. 2) **FUS** (Xia et al. 2022), selecting with a criterion named forgetting score, where the sample with a high forgetting score is utilized to be injected the backdoor. 3) **RD** (Wu et al. 2023), selecting with a criterion named distance score, which chooses the sample away from the decision boundary. 4) **HFE** (Xun et al. 2024), introducing a selection criterion based on high-frequency energy features to identify and enhance the effectiveness of backdoor triggers. Since the lack of open-source code, we reimplemented it ourselves.

5.2 Main Results

We conduct experiments under two aspects of backdooring settings: 1) low poisoned ratio and 2) low manipulation strength ϵ, which reduces the possibility of backdoored samples being detected in the real world.

In Figure 5, we compare IFS with four baseline methods under varying values of manipulation strength ϵ. It can be seen that, for varying values of ϵ, the ASR using our proposal IFS always outperforms that of the other four methods. The improvement brought by IFS is significant, especially in the setting of the minimal ϵ. For example, on ImageNet-10 with the patched attack, the improvement of IFS is more than 5% with $\epsilon = 2$, which provides better ASR while ensuring great stealthiness for the backdoored training set.



(a) Training strategies (b) Models

Figure 7: **Sensitivity analyses** on CIFAR-10 with blended attack ($r = 1\%, \epsilon = 0.1$). (**a**) We search for backdoor samples using VGG16 with IFS, using different training strategies. Then, we train a ResNet-18 with a fixed setting (optimizer: SGD, batch size: 256, Lr: 0.01). (**b**) We fix the training strategy and evaluate the ASR of IFS under different combinations of surrogate and victim models. W: WideResNet28-10, V: VGG16, R: ResNet18, I: Inception-V3

In Figure 6, we evaluate the performance of IFS from another aspect, *i.e.*, the backdoor poison rate r, which is also critical to guarantee stealthiness. It can be seen that on all settings of different backdoor rates r, the performance of our method IFS is consistently better than other methods except on ImageNet-10 and on rad_db with $r = 0.005$. The improvement of IFS is non-trivial. For example, under a very small backdoor rate $r = 0.005$ on CIFAR-10, IFS achieves more than 3% improvement compared with the counterpart state-of-the-art methods.

Figure 8: Comparisons of (a),(b) *the variance of ASR* under different values of ϵ and (c) *the class-wise ASR* on ImageNet-10 when the backdoor rate $r = 1\%$.

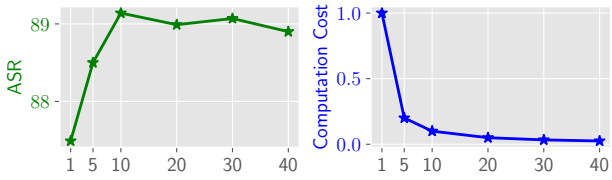


Figure 9: **Hyper-parameter selection** of η on CIFAR-10 with the blended attack ($r = 1\%, \epsilon = 0.1$).

Overall, the significant performance under several settings demonstrates that IFS can attain a high ASR while maintaining high stealthiness (a small value of r and ϵ).

5.3 More Analyses

Sensitivity. Since black-box backdoor attacks are common in real-world applications, we conduct experiments to widely assess the sensitivity of our algorithm in two key factors: different training strategies and different models.

1) In Figure 7a, we evaluate the performance of IFS under different training strategies. We find that while different training strategies slightly impact the ASR, IFS performs best when the training setup matches that of the victim model. This is because influence calculation is sensitive to these factors: different optimizers can lead to convergence at different local minima, and batch size and learning rate can cause variations in training dynamics and convergence rates. Consequently, there exists a slight bias when computing the influence.

2) In Figure 7b, we try the influence of different models on ASR. When the surrogate model is aligned with the victim model, we observe that selected backdoor samples are the most accurate, resulting in the highest ASR. This can be seen as a white-box backdoor attack. Although the victim model is different from the surrogate model, IFS also achieves promising performance, demonstrating the effectiveness of IFS on real-world backdoor attacks.

Variance on class-wise ASR. In Figure 8, we compare IFS with the counterpart FUS on the metric of Variance and visualize the class-wise ASR for further analysis. It can be seen that our proposal significantly reduces the variance on class-level ASR under all manipulation strength settings (see (a) *vs.* (b)). When $\epsilon \geq 0.15$ (the blue line), the variance is very close to 0, meaning that the ASR on almost all classes achieved 100%. In contrast, there exists a variance over class-level ASR for FUS. In subfigure (c), we can observe that IFS achieves more than 50% ASR on all classes. However, in many classes (like the class $4, 8, 9$), the ASR of FUS is no more than 30%. Therefore, in a setting of minimal value of ϵ, IFS can enhance the attack success rate on many classes, promoting the real-world application of IFS.

Component		ImageNet-10		Raf_db	
Group split	Fair sel.	*ASR*	*Cost*	*ASR*	*Cost*
✗	✗	$64.49_{\pm 1.0}$	$\times 1$	$79.94_{\pm 1.2}$	$\times 1$
✗	✓	$67.34_{\pm 0.6}$	$\times 1$	$81.89_{\pm 0.7}$	$\times 1$
✓$_{(\eta=10)}$	✗	$68.94_{\pm 0.5}$	$\times \frac{1}{10}$	$83.20_{\pm 0.3}$	$\times \frac{1}{10}$
✓$_{(\eta=10)}$	✓	$70.94_{\pm 0.5}$	$\times \frac{1}{10}$	$84.94_{\pm 0.5}$	$\times \frac{1}{10}$

Table 1: **Ablation studies** about the effectiveness of each component in IFS. We adopt the blended attack with $r = 1\%, \epsilon = 0.1$.

5.4 Ablation Studies

Hyper-parameter. In Figure 9, we examine the sensitivity of IFS to varying values of η, the sole hyper-parameter in IFS, which significantly impacts computational cost. We observe that when η is set to 1 where $D_{\text{sub}} = D_N$, the ASR is at its lowest while the computational cost is the highest. As η increases, the computational cost gradually decreases. For instance, setting $\eta = 10$ reduces the computational cost to just $\frac{1}{10}$ of the original. When the value of η is in the interval of $[10, 40]$, our algorithm is insensitive to the variation of η. Ultimately, we select $\eta = 30$ for CIFAR-10, as it provides strong performance and minimal computational cost.

Effectiveness of each component. In Table 1, we investigate the effectiveness of each component in IFS, including the group split strategy and influence-based fair selection strategy. We can see that when we compute the influence over all training samples (*w/o* group split and class fair selection), the performance of IFS is worst while the computation cost is highest. When we add these two components into the base setting independently, IFS can achieve better performance consistently. We can adjust the parameter η in the group split to achieve a fast computation efficiency. The best performance (70.94% on ImageNet-10 and 84.94% on Raf_db) is achieved when both two components are adopted.

6 Conclusion

In this paper, we investigate sample-discriminative backdoor sample selection and uncover the issue of unfair selection across different classes under minimal manipulation strength. To address this, we propose an influence-based fair selection method (IFS), ensuring that an equal number of instances are selected from each class based on the influence of training samples with triggers on ASR. IFS is computation-efficient, utilizing a group-based pruning strategy to avoid calculating influence across all samples. Extensive experiments on varying black-box settings verify that our proposal IFS consistently achieves the best performance, highlighting its effectiveness in real-world applications.

Acknowledgements

This research is supported by the Ministry of Education, Singapore, under its MOE AcRF Tier 3 Award MOE-MOET32022-0006. This research is also supported by the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Programme (AISGAward No: AISG2-GC-2023-009).

References

Agarwal, N.; and Bullins, B. 2017. Second-order stochastic optimization for machine learning in linear time. *JMLR*.

Bagdasaryan, E.; Veit, A.; Hua, Y.; Estrin, D.; and Shmatikov, V. 2020. How to backdoor federated learning. In *AISTATS*.

Chen, X.; Liu, C.; Li, B.; Lu, K.; and Song, D. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv:1712.05526*.

Cinà, A. E.; Grosse, K.; Vascon, S.; Demontis, A.; Biggio, B.; Roli, F.; and Pelillo, M. 2021. Backdoor learning curves: Explaining backdoor poisoning beyond influence functions. *arXiv:2106.07214*.

Cook, R. D. 2000. Detection of influential observation in linear regression. *Technometrics*.

Cook, R. D.; and Weisberg, S. 1980. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*.

Gu, T.; Liu, K.; Dolan-Gavitt, B.; and Garg, S. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*.

He, P.; Xu, H.; Xing, Y.; Ren, J.; Cui, Y.; Zeng, S.; Tang, J.; Yamada, M.; and Sabokrou, M. 2023. Confidence-driven Sampling for Backdoor Attacks. *arXiv:2310.05263*.

Jia, J.; Liu, Y.; and Gong, N. Z. 2022. Badencoder: Backdoor attacks to pre-trained encoders in self-supervised learning. In *IEEE S&P*.

Jiang, W.; Li, H.; Xu, G.; and Zhang, T. 2023. Color backdoor: A robust poisoning attack in color space. In *CVPR*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *PNAS*.

Koh, P. W.; and Liang, P. 2017. Understanding black-box predictions via influence functions. In *ICML*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Li, S.; Deng, W.; and Du, J. 2017. Reliable Crowdsourcing and Deep Locality-Preserving Learning for Expression Recognition in the Wild. In *CVPR*.

Li, X.; Chen, Z.; Zhao, Y.; Tong, Z.; Zhao, Y.; Lim, A.; and Zhou, J. T. 2021. Pointba: Towards backdoor attacks in 3d point cloud. In *ICCV*.

Li, Z.; Sun, H.; Xia, P.; Li, H.; Xia, B.; Wu, Y.; and Li, B. 2024. Efficient Backdoor Attacks for Deep Neural Networks in Real-world Scenarios. In *ICLR*.

Li, Z.; Sun, H.; Xia, P.; Xia, B.; Rui, X.; Zhang, W.; Guo, Q.; and Li, B. 2023a. A Proxy Attack-Free Strategy for Practically Improving the Poisoning Efficiency in Backdoor Attacks. *arXiv e-prints*, arXiv–2306.

Li, Z.; Xia, P.; Sun, H.; Zeng, Y.; Zhang, W.; and Li, B. 2023b. Explore the effect of data selection on poison efficiency in backdoor attacks. *arXiv:2310.09744*.

Liang, J.; Liang, S.; Liu, A.; Jia, X.; Kuang, J.; and Cao, X. 2024. Poisoned forgery face: Towards backdoor attacks on face forgery detection. *arXiv:2402.11473*.

Nguyen, Q. H.; Ngoc-Hieu, N.; Ta, T.-A.; Nguyen-Tang, T.; Wong, K.-S.; Thanh-Tung, H.; and Doan, K. D. 2024. Wicked Oddities: Selectively Poisoning for Effective Clean-Label Backdoor Attacks. *arXiv:2407.10825*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*.

Wang, Z.; Zhu, H.; Dong, Z.; He, X.; and Huang, S.-L. 2020. Less is better: Unweighted data subsampling via influence function. In *AAAI*.

Wenger, E.; Passananti, J.; Bhagoji, A. N.; Yao, Y.; Zheng, H.; and Zhao, B. Y. 2021. Backdoor attacks against deep learning systems in the physical world. In *CVPR*.

Wu, Y.; Han, X.; Qiu, H.; and Zhang, T. 2023. Computation and Data Efficient Backdoor Attacks. In *ICCV*.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*.

Xia, P.; Li, Z.; Zhang, W.; and Li, B. 2022. Data-efficient backdoor attacks. In *IJCAI*.

Xiang, Z.; Miller, D. J.; Chen, S.; Li, X.; and Kesidis, G. 2021. A backdoor attack against 3d point cloud classifiers. In *ICCV*.

Xun, Y.; Jia, X.; Gu, J.; Liu, X.; Guo, Q.; and Cao, X. 2024. Minimalism is King! High-Frequency Energy-based Screening for Data-Efficient Backdoor Attacks. *IEEE TIFS*.

Yang, S.; Xie, Z.; Peng, H.; Xu, M.; Sun, M.; and Li, P. 2022. Dataset pruning: Reducing training data by examining generalization influence. In *ICLR*.

Zhu, Z.; Zhang, M.; Wei, S.; Shen, L.; Fan, Y.; and Wu, B. 2023. Boosting backdoor attack with a learnable poisoning sample selection strategy. *arXiv:2307.07328*.